УДК 004.8.032.26

СРАВНИТЕЛЬНЫЙ ОБЗОР «БЫСТРОГО» АЛГОРИТМА ГЕНЕРАЦИИ ГРАФОВ

Кочурко В.А.
УО «Брестский государственный технический университет», г. Брест

Введение. Задача создания полного списка регулярных графов, вместе с задачей распознавания изоморфных графов, является одной из старейших задач конструктивной комбинаторики. Помимо простого научного интереса, генерация таких графов имеет также и практическое значение.

Регулярные графы применяются в разнообразных областях, например, в peer-to-peer сетях, некоторые из которых базируются на связных регулярных неориентированных графах, при идентификации молекул веществ, при разработке баз знаний интеллектуальных систем или при создании чипов и микросхем.

В данной работе анализируется так называемый «быстрый» алгоритм, предложенный профессором Баварского университета Маркусом Мерингером в [1], и сравнивается с алгоритмом двух вершин, предложенным в [2]. Анализ базируется на программных реализациях этого алгоритма, созданном коллегами господина Мерингера, и алгоритма двух вершин, созданном в Брестском государственном техническом университете.

Постановка задачи. Регулярный граф $\tilde{A}_N^{\hat{E}}$ - граф, у которого степени всех вершин равны между собой и где ***N*** – количество этих вершин, а ***K*** – степени вершин. Задача состоит в генерации всех неизоморфных между собой регулярных графов с заданными ***K*** и ***N***.

Переформулируем задачу в терминах теории групп. Пусть G_N - множество простых помеченных графов с множеством вершин ***{1, …, n}***. Подмножество из ***K***-регулярных графов обозначим $R_{N,K}$. Опишем граф через множество его ребер:

$$\tilde{A} = \{e_1, e_2, ..., e_t\} \subseteq \binom{\{1, ..., n\}}{2} =: X_n$$

Если $e = (v, w) \in X_n$ обозначает ребро, то всегда принимается, что ***v < w***. Симметрическая группа S_N действует на основе множества X_N и, тем самым вызывает действия на G_N и $R_{N,K}$.

$S \,//\, G_N$ и $S \,//\, R_{N,K}$ обозначают орбиты этих действий. Задача состоит в вычислении набора представителей орбиты $S \,//\, R_{N,K}$.

Краткий обзор «быстрого» алгоритма. Для начала упорядочиваем полное множество рёбер X_N следующим образом: Для $e = (v, w), e' = (v', w') \in X_n$ определим отношение «больше» - $e < e' := v < v' \vee (v = v' \wedge w < w')$.

Тогда, пользуясь этим отношением, упорядочим все ребра в графах ***Г***. Аналогично, упорядочим все графы ***Г*** в множестве графов G_n, пользуясь отношением «больше» для графов:

$$\tilde{A} = \{e_1, e_2, ..., e_t\},\ \tilde{A}' = \{e_1', e_2', ..., e_{t'}'\}$$

$$\tilde{A} < \tilde{A}' :=$$

$$(\exists i \le \min\{t, t'\} : e_j = e_j' \forall j < i \wedge e_i = e_i')$$

$$\vee\ (t < t' \wedge e_j = e_j' \forall j \le t)$$

В статье [1] показано доказательство необходимости поиска канонических представителей - $rep_<(S//R_{N,K})$ - орбиты $S//R_{N,K}$ как минимальные в их орбите. Так же доказана теорема, по которой, если граф ***Г*** является представителем минимальной орбиты, то и граф ***Г₁***, где $\tilde{A}_1 < \tilde{A}$ и $\tilde{A}_1 \subset \tilde{A}$, также является представителем минимальной орбиты. Базируясь на этой теореме, формулируем рекурсивный алгоритм перебора на псевдоязыке - процедура ***P(Г)***:

1. Точка входа: ***P***({1,2}).

2. Проверка: может ли входной аргумент – граф ***Г*** – быть распространён на ***K***-регулярный граф с ***N*** вершинами, если нет – выход.

3. Проверка: выполняется ли условие $\tilde{A} \in rep_<(S // R_{N,K})$, если нет – выход.

4. Если $\tilde{A} \in R_{N,K}$: добавить ***Г*** в выходное множество, выход.

5. Для каждого $e \in X_n$ с $e > \max\{ e' \in \tilde{A} \}$ вызывают ***P(Г+{e})*** с увеличением ***e***.

Очевидно, что трудоёмкость и используемая память алгоритма в таком виде несравнимо более велики, чем для алгоритмов, описанных в [2].

Уменьшение этих параметров происходит за счёт использования полуэмпирических лемм Грюнда и Бринкманна, которые подробно рассмотрены в [1].

Используемая память:

$$O\left(\left(\frac{N*(N-1)}{2}\right)^{\frac{NK}{2}}\right) \Rightarrow O(N^{NK})$$

Краткий обзор алгоритма двух вершин. Алгоритм двух вершин был подробно рассмотрен и сравнен с другими алгоритмами в [2] и является, возможно, лучшим представителем семейства алгоритмов, базирующихся на обработке исключительно матриц смежности. Его трудоёмкость меньше, а список генерируемых графов полон.

Допустим, что существует исходный граф $\tilde{A}_n^k$, из которого необходимо получить граф $\tilde{A}_{n+2}^k$. Тогда возьмем ребро **(x_i, x_j)** этого графа, и установим на нем две вершины x_{n+1} и x_{n+2}. Эти вершины будут смежны между собой и смежны вершинам x_i и x_j.

Вторым шагом является удаление ребер. Поочередно перебираются все ребра, неинцидентные вершинам x_i и x_j, и в каждом случае такое ребро **(x_{i1}, x_{j1})** удаляется. В таком случае появляется возможность установки новых ребер, причем двумя способами: **(x_i, x_{j1})** и **(x_{i1}, x_1)**, или же **(x_{i1}, x_i)** и **(x_j, x_{j1})**. Полный перебор всех ребер в качестве входных завершает данный алгоритм.

Трудоемкость данного алгоритма:

$$\frac{(M-2)N(N-1)}{2} \Rightarrow O(MN^2)$$

Количество используемой памяти:

$$O\left((K*N)^{K-1}\right) \Rightarrow O((KN)^K)$$

Программное сравнение. Главной сложностью в попытке сравнения алгоритмов является сложность определения трудоёмкости быстрого алгоритма из-за использования большого количества исключений и нечёткого обозначения алгоритма проверки принадлежности графа множеству канонических представителей; следовательно, сравнению по главному количественному критерию – трудоёмкости – алгоритмы не подлежат. Сравне-

ние же верхних оценок используемой памяти даёт поверхностное представление о чуть большей компактности алгоритма двух вершин. Однако попробуем сравнить результаты программных реализаций этих алгоритмов, для чего приведём таблицу таковых:

N	K	Кандидаты (Быстрый)	Кандидаты (2х вершин)	Время, с (Быстрый)	Время, с (2х вершин)
10	3	37	84	0	0
12	3	214	672	0	0,3
14	3	1406	7208	0,1	2
16	3	10432	49874	1	18,3
18	3	96279	-	10,8	-
20	3	1079585	-	139	-

Испытание проводилось на одинаковой компьютерной конфигурации. Графа «Кандидаты» для каждого алгоритма – это количество рабочих графов перед последней стадией отсеивания. Также стоит упомянуть, что на выходе быстрого алгоритма – список неизоморфных между собой графов, в то время как к выходному списку алгоритма двух вершин следует применить ещё и любой алгоритм изоморфного отсеивания, что увеличивает в разы трудоёмкость и, соответственно, время работы.

Заключение. Таблица 1 ясно указывает на сильный перевес в производительности для быстрого алгоритма, в то время как единственным его слабым местом является верхняя оценка количества используемой памяти. На общую оценку алгоритма в наше время больших объёмов памяти это уже не влияет так существенно, как могло бы ещё лет 20 назад – алгоритм оправдывает своё название.

Литература

1. Meringer, M. Fast Generation of Regular Graphs and Construction of Cages. Journal of Graph Theory 30, 137-146, 1999.

2. Кочурко, В. А. Некоторые алгоритмы генерации графов. Сборник конкурсных научных работ студентов и магистрантов / В.А. Кочурко – БрГТУ, 2009. – Ч. 1. – С.163-167.

3. Шуть, В. Н. Генерация регулярных связных графов / В.Н. Шуть, В.М. Свирский, Г.Л. Муравьев, С.В. Анфилец / Вестник БрГТУ. – 2006. – №5: Физика, информатика, математика. – С. 44 – 47.

УДК 372.016:51

ОСОБЕННОСТИ ИСПОЛЬЗОВАНИЯ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ ДЛЯ АДАПТАЦИИ УРОВНЯ МАТЕМАТИЧЕСКОЙ ПОДГОТОВКИ СТУДЕНТОВ-ИНОСТРАНЦЕВ

Крагель Е.А.

УО «Брестский государственный технический университет», г. Брест

Многовекторность внешней политики Беларуси делает ее привлекательной с точки зрения инвестиций и не только материальных. В настоящее время наметилась тенденция увеличения числа иностранцев, обучающихся в отечественных вузах. Последнее обусловлено следующими факторами:

– стабильная политика социальной ситуации в стране (отсутствие конфликтов на межнациональном и религиозном уровне);

– низкая стоимость обучения (оплата за год обучения в вузе для иностранцев превышает в несколько раз плату за обучение отечественных студентов, тем не менее эта сумма приемлема по сравнению с общемировыми стандартами);